The Use of Artificial Intelligence Techniques to Improve the Multiple Payload Integration Process

Dannie E. Cutts [1]
Space Operations Department
Teledyne Brown Engineering
Huntsville, Alabama

Brian K. Widgren
Manager, CDMS Integration
Payload Integration Engineering Department
Teledyne Brown Engineering
Huntsville, Alabama

Abstract

A maximum return of science and products with a minimum expenditure of time and resources is a major goal of mission payload integration. A critical component then, in successful mission payload integration is the acquisition and analysis of experiment requirements from the Principal Investigator (PI) and Payload Element Developer (PED) teams. This paper describes one effort to use Artificial Intelligence (AI) techniques to improve the acquisition and analysis of experiment requirements within the Payload Integration Process.

[1] The authors may be contacted at:
Teledyne Brown Engineering, MS-172
300 Sparkman Drive NW
Huntsville, Alabama 35807-7007
(205)726-5929

OVERVIEW

As the payload integration contractor to Marshall Space Flight Center for Spacelab payloads, Teledyne Brown Engineering (TBE) has been heavily involved in the acquisition, analysis, and integration of payload requirements for a number of years. NASA/MSFC and TBE are currently involved in efforts to streamline and improve the mission integration process. Part of this improvement effort involves the use of Expert Systems in several areas. A number of benefits are anticipated from the use of these systems, including:

- A better understanding by the PI/PED teams of STS and Spacelab capabilities,
- On-line help and documentation capabilities,
- On-line data validation rules to check data for accuracy and reasonableness,
- A more consistent approach to experiment requirements gathering and analysis,
- Increased quality in the requirements definition data provided to the integration team, resulting in fewer iterations between the PI/PED and integration teams,
- Increased quality in the analyses performed on those requirements,
- Reduction in the need for members of the integration team to travel to PI/PED sites,
- Retention and documentation of corporate expertise in payload integration, and
- Training tools for Payload Integration Engineers.

Throughout the Spacelab integration process, averaging about 42 months from Authority to Proceed (ATP) to the actual mission, a series of readiness reviews are held to ensure that requirements definition and integration are progressing on schedule. The current approach to gathering experiment requirements is human intensive requiring numerous iterations between the PI/PED/Integration teams [FIG 1].

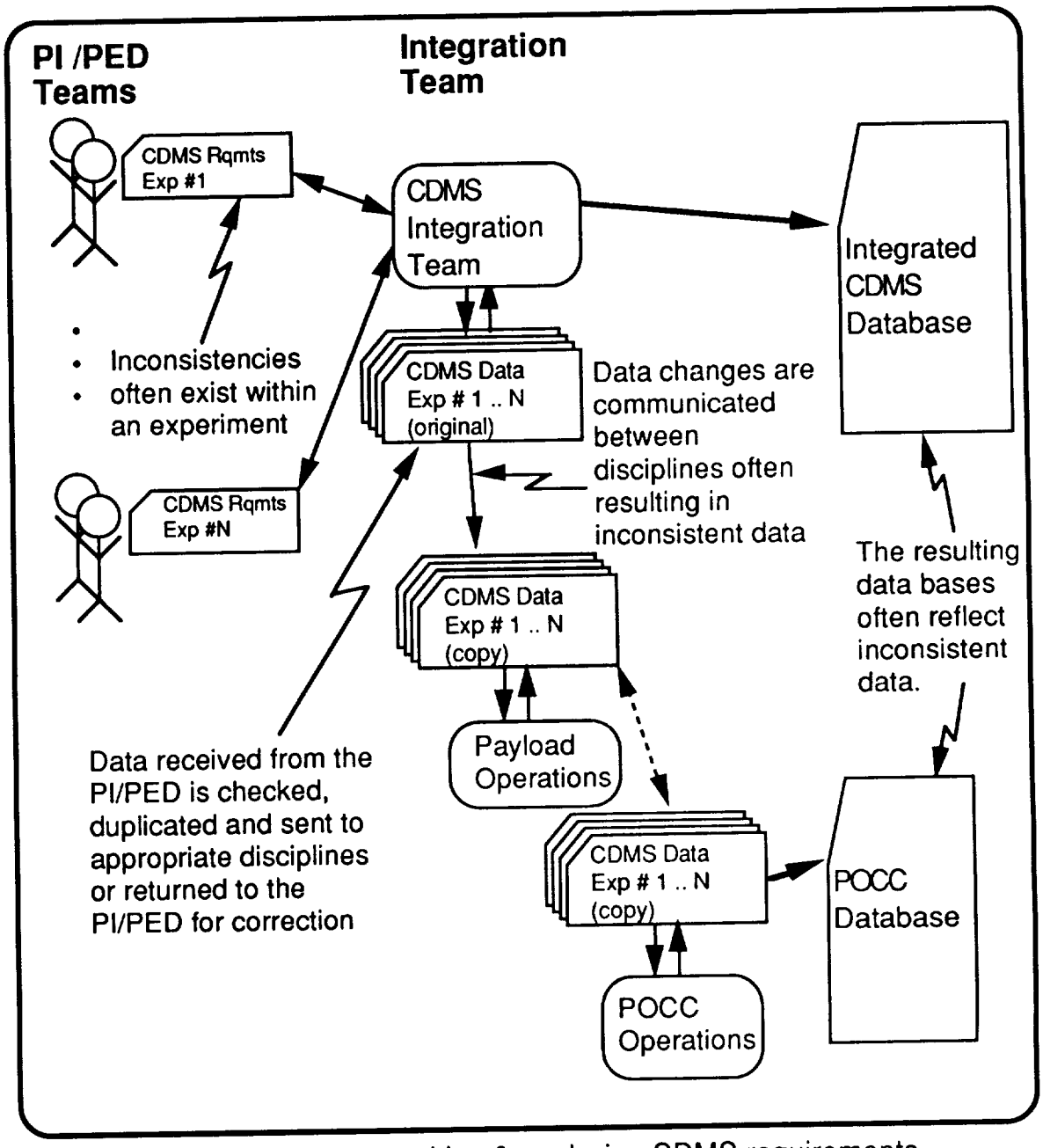
Much of this iteration takes place as a result of changing requirements. A volumous "data pack" of information describing STS and Spacelab capabilities and constraints is sent to each PI/PED. Instructions outlining responsibilities and deadlines are included in the pack along with directions for completing the requirements definition forms. Digesting and interpreting this data pack can present a formidable task for the PI/PED. As a result, requirements are often submitted late, poorly defined, or incomplete. All these situations can significantly affect the readiness reviews and thus the entire integration process.

This paper describes one project within TBE to use Expert Systems to automate portions of the acquisition and analysis effort for the Commanding and Data Management system (CDMS) requirements. The acquisition and analysis of CDMS requirements begin very early in the mission design process and impact nearly every discipline. Historically, changes to CDMS requirements take place throughout the entire life of the mission design cycle. These changes come about for various reasons including users not understanding Spacelab or STS capabilities, changing hardware/software configurations, design reviews, operational considerations as well as problems resulting from the complex interactions between an individual experiment and the integrated payload. The high cost of incorporating changes late in the integration process made the CDMS expert system a "high payback" candidate for development.

CDMS DESCRIPTION

CDMS data consists of a set of PI/PED generated documents defining requirements for on-board data processing and display, telemetry monitoring and experiment commanding by the Spacelab experiment computer as well as downlink requirements and POCC (Payload Operations Control Center) processing requirements.

The POCC supports extensive data processing capabilities including ground monitoring of telemetry data, acquisition and storage of science data, as well as ground commanding of the on-board experiments by the PI.



Current approach to acquiring & analyzing CDMS requirements

[FIG 1]

These requirements are defined in a set of eight tables generated by the PI/PED team and expanded by the integration team and other contractors.

CDMS REQUIREMENTS ACQUISITION

Early in the mission design process, PI/PED teams are required to supply preliminary CDMS requirements. Since the initial submission of these requirements takes place very early in the design phase, some

requirements may not be firm because of unresolved hardware or software issues. The fact that these requirements remain unresolved may or may not be identified in the submitted documents. The integration team may have no way of knowing which requirements are to be supplied later. Often these "missing" requirements are not identified until late in the integration process resulting in reworks or delays. With the automated acquisition tool, PI/PEDs can flag data as "TBS" so that the integration team can follow up on later definition.

Under the current approach for gathering CDMS requirements, the PI/PED supplies requirements in paper format to the CDMS engineers. These engineers then do a manual cross checking of the paper inputs to ensure accuracy and agreement across tables before inputting it into an electronic format. Some errors in the data are missed during the "paper analysis" while others occur as a result of operator entry errors. Errors identified in the PI/PED supplied requirements are returned to the PI/PED for revision. Submission of better initial data from the PI/PED will help to shorten the number of these iterations that take place.

These CDMS data are currently stored as text files on either VAX or Macintosh computers. Various analyses (both manual and automated) are performed on these text files and modifications are made directly to them. After analyses are performed on individual files representing experiments, an integrated CDMS "database" file is constructed by concatenating the individual experiment files. Analyses are then performed on the complete file representing the integrated payload.

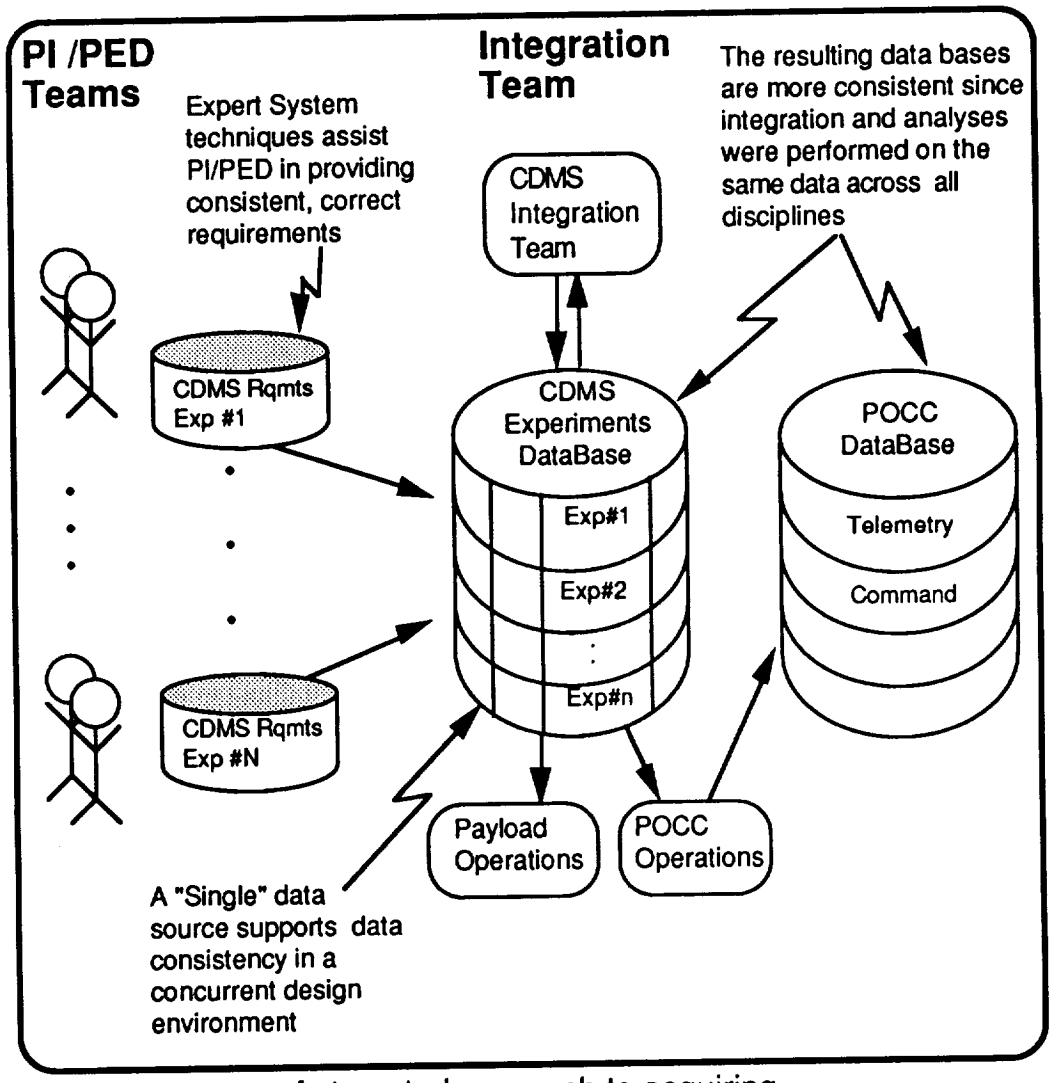
In an effort to improve the above process, we plan to have the PI/PED teams submit CDMS requirements electronically using an acquisition tool [FIG 2] being developed by TBE .

This tool utilizes a number of expert system techniques including limited "object" support, constraint checking, and a simple form of retraction in which data is valid only when supported by other data (Doyle 1981). When the supporting data changes, the "supported" assertion is changed. These "dependency" relations are implemented in a hypertext environment in which data objects contain lists of other data objects on which they are dependent and which are dependent on them. This "dependency" matrix is used to guide the dialogue with the user, provide explanations and support data retraction.

Some problems in the current data requirements definition occur as a result of the user not understanding STS or Spacelab capabilities. To assist in this area, the automated acquisition tool offers explanations of the questions being asked which can include detailed descriptions of hardware or carrier issues. In addition, we have implemented a "why" facility which can provide context sensitive information on why the particular datum is important. This capability is closely tied to the "dependencies" relations mentioned earlier. This acquisition system will assist the PI/PED teams in supplying complete and correct requirements definitions to the integration team.

CDMS REQUIREMENTS ANALYSIS

The second component of this process improvement effort deals with the analysis of the acquired CDMS data. Once the PI/PED generated requirements documents reach the integration team, they must be checked independently for problems and then combined with other experiment requirements files to perform integrated analyses. Error checking in the automated requirements acquisition system described above ensures that the data arrives relatively error free. Since the documents are submitted electronically by the PI/PED teams, errors introduced during the data input phase are eliminated. Engineers are also freed from the task of inputting the PI/PED data. The current text file storage of CDMS requirements data is being replaced with a relational database. Having these data stored in a relational database allows the integration team to more easily manipulate the CDMS data.

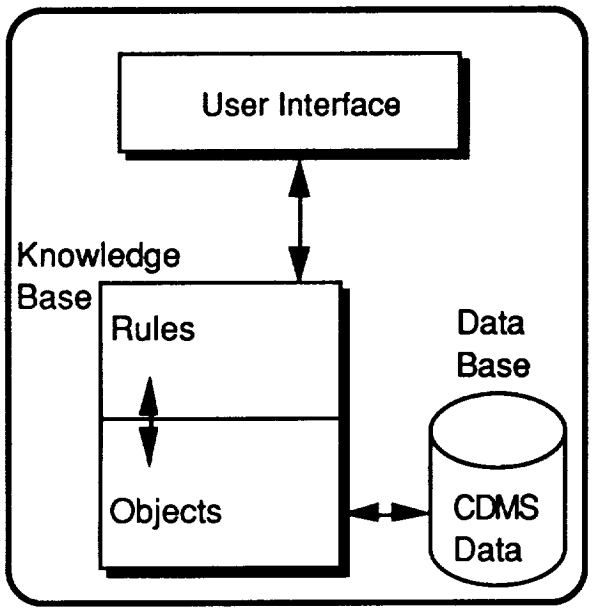


Automated approach to acquiring and analyzing CDMS requirements

[FIG 2]

As these analyses progress, the integration team adds a significant amount of mission dependent data. The data supplied by the integration team is also automatically checked as it is entered by the engineers.

As mentioned earlier, CDMS data impacts numerous other disciplines within the integration team. In the past, a significant amount of time was spent by the CDMS group in inputting and checking PI/PED data before that data could be made available to other disciplines. Under the automated acquisition and analysis approach, the data arrives in electronic format with significant data checking already performed. As a result, portions of the data can be made available to other disciplines much faster. This analysis tool is used to transfer the PI/PED requirements from the electronic file format submitted by the PI/PED to a relational database. The user interface of the analysis tool also allows the integration team to manipulate the data and generate various report formats. Rules are used to access the data stored in the relational database to perform validity checks within and across experiments [FIG 3]. Both mission dependent and mission independent checks are made.



Relationship between User, Domain Knowledge & Data

[FIG 3]

REQUIREMENTS DATABASE

In support of this automated acquisition and analysis effort, TBE is establishing an integrated "requirements database" using Oracle. This integrated requirements database will provide a single source of experiment data from which all members of the integration team will work. This will eliminate the problem of different teams working with different versions of the same data. Impacted users can be automatically notified when data has been changed. We also are planning to use the integrated database to generate portions of deliverables (including documentation) for use within TBE and NASA as well as other subcontractors. It is anticipated that some of the applications accessing the integrated requirements database will utilize expert systems while others will be conventional systems.

METHODOLOGY

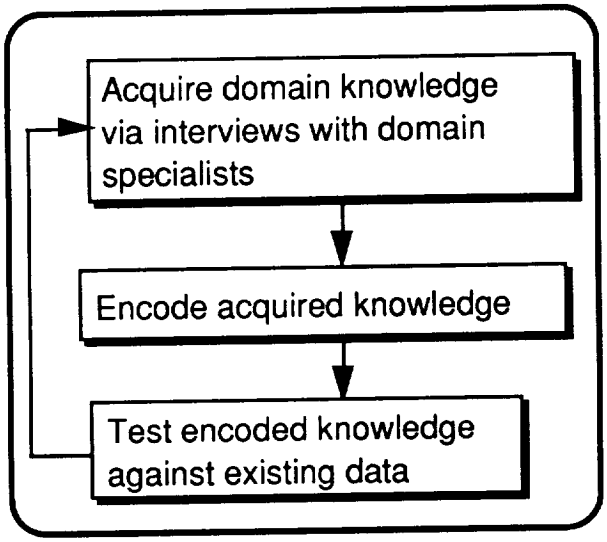
In the design and development of both the acquisition and analysis applications a number of issues arose. They are listed and briefly discussed below:

1. Knowledge Acquisition

Most experts agree that knowledge acquisition is one of the most (if not THE most) critical components in the development of knowledge based systems - and often the most labor intensive (Gaines 1989). Although tools exist for the automated acquisition of domain knowledge, we chose to use manual interview techniques for this effort since the knowledge acquisition task in this case was relatively straightforward. A great deal of knowledge about data relationships is already documented in a set of detailed instructions to the PI/PED supplying CDMS requirements (MDC 1991). This document covers only the on-board CDMS data requirements (four of the eight tables populated by the PI/PED), but did provide an excellent starting point for acquiring domain knowledge. There were several reasons why this document provided only a starting point, and not the entire knowledge base. First, the document only defines data relationships and constraints. This type of knowledge, while important, does not support any type of intelligent dialogue between the system and the PI/PED user. Also missing were heuristics and explanation knowledge of how a human expert would query the PI/PED to acquire CDMS data. This control knowledge will be discussed later.

Since the PI/PED customarily provides paper copies of CDMS requirements and no human is usually involved in the initial acquisition of CDMS data, no documentation of the acquisition process existed. As a result, interviews were held with CDMS domain experts in which they started with the rules coded from the CDMS instruction document (MDC 1991), and then added control knowledge and explanation knowledge to the rules to provide a dialogue structure for the acquisition system to follow in gathering CDMS requirements from the PI/PED. (Craig 1990) describes a relationship between domain and control knowledge which we largely followed in this effort. [FIG 5] is taken from that reference.

A conventional interview technique was followed in which the knowledge engineer interviewed the domain specialist (often using the partially completed tool), encoded the acquired knowledge, and then supplied the updated system to the domain specialist for further testing [FIG 4]. (Gruber 1987) presents three principles of design for knowledge acquisition systems. Although that paper discusses automated acquisition approaches, the principles are worth noting. The first principle was that the knowledge engineer should provide a "language of task-level terms natural to the expert yet sufficient to solve the problem". In this application, knowledge of how to validate the data was already expressed to a large extent as rule structures. The second principle to be followed was that "representational primitives should be explicit and able to stand alone". Again, this posed no real problem since the existing instructions for requirements definition were already stated in an "IF-THEN" format. The third principle dealt with "avoiding generalizations except when necessary". We found that when the domain expert attempted to generalize, we often encountered new knowledge in the form of "exceptions to the rule". In this application, attempts to generalize often helped elicit knowledge.



Elicit-Encode-Test cycle
[FIG 4]

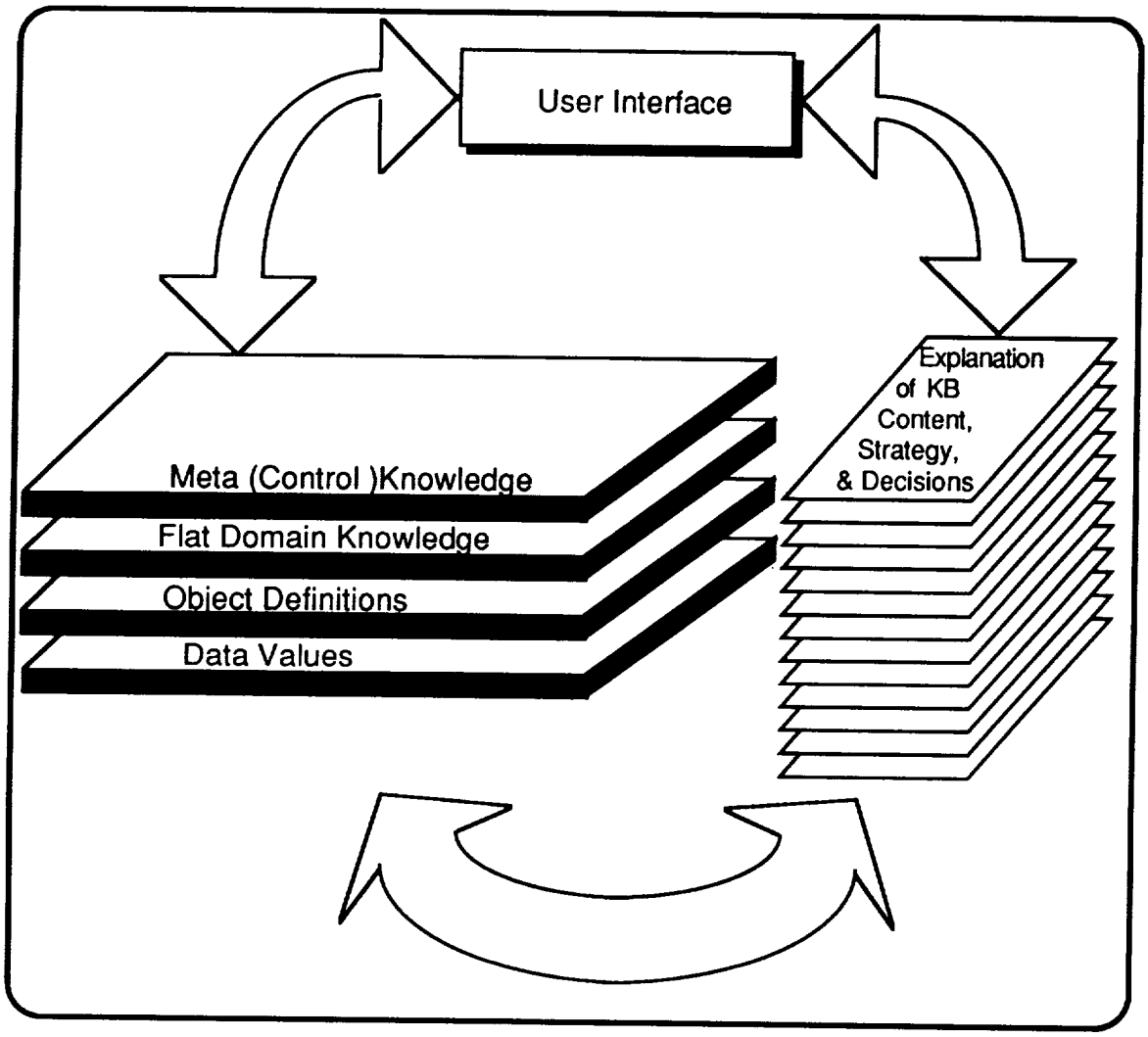
As the interviews were completed, the developer encoded the rules acquired during the interview. The tool was then made available to the domain experts for testing. One result of this incremental "ELICIT - ENCODE - TEST" development approach was that we were able to start using portions of the analysis system early in the development cycle. Another result was that customer confidence in and expertise with the tool grew during development.

2. Knowledge Representation

(Kitto 1989) points out that a failure to map properly between the Knowledge Acquisition technique, Knowledge Acquisition tool, Knowledge Representation methodology and the problem type will likely cause the effort to fail. During the domain expert interviews, most knowledge was structured as "IF-THEN" statements. This led to the use of a rule-based representation for domain knowledge, with the underlying parameters modeled as objects. The mapping between the Knowledge Acquisition technique and the Knowledge Representation paradigm was very straightforward and allowed us to model the domain naturally. In addition, the knowledge encoded in the knowledge base was easy to understand and maintain.

3. User Interface

User acceptance of a system is highly dependent on an appropriate human-computer interface. This interface must be responsive to a range of user abilities. In this effort, we have made no attempt to build user models to account for various ability levels. Our approach has been to provide information at a relatively high level, but to provide help in questions asked of the user and by providing explanations upon request. Initial experiences with the automated analysis tool indicate that this approach is sufficient. (Wexelblat 1989) gives an excellent overview of characterizations of users by ability level while (Swigger 1989) addresses research issues in human-computer interfaces for tutoring systems.



Relationship between knowledge, data and explanation

[FIG 5]

The interface to the automated acquisition tool had to provide PI/PED teams with an easy to use "point and click" interface which could provide context sensitive help when appropriate. The tool was designed to assist the PI/PED user in "constructing" CDMS requirements. The interface to the automated acquisition tool is designed to query the user for requirements definition. As the user is asked questions about parameter definitions, help is provided and constraint information is available. The queries are formed by inserting context sensitive information into a text string. For example, if a constraint existed between two data items A and B such that A has to be less than B, and if A had the value of 30, the query might look something like.. *Provide a value for B that is greater than 30*... If the user then asked for an explanation, he would be told about the relationship between A and B and the existing value for A. At that point, he could choose to supply B or modify A.

The interface to the automated analysis tool to be used by the integration team was designed to look much like the data formats the engineers were accustomed to. Less explanation is supplied and less control is exercised by the system. Both these tools were implemented in a hypertext tool (SuperCard) which allowed users to move through the data in an unconstrained fashion.

The requirements acquisition tool was somewhat more constrained than the analysis tool because it uses the dependencies relationships between the data items to guide the dialogue between the user and the system.

4. Dialogue Control

Dialogue control in the requirements acquisition tool was implemented using the dependency matrix mentioned earlier. The acquisition process begins with a leadoff series of questions, which are then used to guide or constrain further questioning. Also, much support information is provided in the query itself. This helps the user to understand the significance of the question. For example, if the user specifies that the parameter being defined requires on-board displays, the system then queries for further information on the display requirements and reminds the user that he had earlier provided the requirement for displaying the parameter. If the user indicates that the parameter will not be displayed on-board, he is not asked for unnecessary information - a situation that cannot be avoided in the paper CDMS requirements forms. This control, however, does not prevent the user from modifying earlier definitions. The details of how this is accomplished is discussed in the next section.

5. Consistency Maintenance

One of the problems with the old paper requirements approach is that users often have a need to modify requirements. As in the case given above, if a user wants to retract the requirement for on-board display of a parameter, all display requirements data for that parameter must be withdrawn. With the paper approach, relations between data across tables cannot be linked in such a way that data changed in one table changes all its associated data. In the automated acquisition system, however, these relations are modeled in a dependencies matrix so that when a datum changes, rules are triggered that modify all the associated data. This approach to consistency maintenance is loosely based in Doyle's justification-based truth maintenance system (Doyle 1981).

6. Explanation

(Craig 1990) and (Fennel 1990) point out that in many rule-based expert systems, explanation and why facilities are implemented using the rule firing chain to trace each step of the inference process. They point out that this approach is often not appropriate for providing meaningful explanations since the rule tracings often contain inferences at the wrong level of abstraction. They implemented a layered control architecture [FIG 5] in which explanations can be provided on various levels (i.e. explanation of what a datum represents, constraint knowledge about that datum, etc.) This structure was largely followed in developing the explanation system.

(Clancey 1988) identifies four categories of explanation knowledge closely related to those described above:

a. Heuristic Rules which, identify relations between data and rules using that data,
b. Structure knowledge, which identifies dependency relations among data,
c. Strategy knowledge, which identifies the procedure for applying rules, and
d. Support knowledge, which provides the justification for rules.

The dependency matrix (which identifies dependencies between data elements) is used by the explanation facility to provide justification for why a particular datum is needed. Explanation about a particular datum is statically defined either in textual or graphical format and is provided to the user upon request during a session.

IMPLEMENTATION

This Expert System is currently hosted on a Macintosh Computer using Nexpert Object for knowledge representation and inference control, Oracle for data storage, and SuperCard for the interface. The availability of commercial bridges between all these applications made interfacing them straightforward. Although some inquiries have been made as to the possibility of

rehosting the software on other platforms, no work has begun in this area. Both the knowledge base constructed in Nexpert and the data stored in Oracle will transfer to a wide range of platforms. SuperCard runs on the Macintosh platform only, but other interface tools are being considered.

FUTURE DIRECTIONS

As well as improving the existing payload integration process for Spacelab, this technology is also applicable to other aerospace applications including the Space Station Freedom program. We expect many Spacelab experiments to transition to Space Station and anticipate that having many of the experiment requirements acquired, stored in a database, and analyzed will improve the integration process within the Space Station program. Work is underway to identify which portions of the data are carrier independent.

The requirements database is expected to assist in experiment reflights aboard Spacelab where the bulk of the requirements for an experiment do not change. Tools such as these may also be used to assist mission designers in selecting payloads based on mission characteristics. For example, if a microgravity mission is being considered, designers might access the requirements database to identify candidate experiments, and then use the list of candidates to construct optimum payload configurations.

We mentioned earlier that one of the expected benefits of these systems was to use them as training tools. While the systems as currently implemented have no tutoring capabilities, engineers using them can gain a better understanding of their domain. One enhancement in future versions might be to employ tutoring strategies so that the system could be used as a teaching tool.

CONCLUSION

The PI/PED acquisition system described in this paper is planned for use on the WISP-HF/ATLAS-4 mission. WISP (a Canadian experiment) represents the first totally new experiment planned for Spacelab.

Portions of the CDMS analysis tool are currently being used by the integration team on several missions and benefits are already being realized. Maintenance of the CDMS data and the expansion of requirements definitions are much easier in the automated system, and the system has already identified a number of errors in CDMS data for missions under development.

One problem encountered during the development of the analysis tool was no access to a network server version of Oracle within TBE. All development had been done on a Macintosh using a "stand alone" Oracle. The CDMS table structures had been defined and populated with existing mission data. However; the availability of the data was limited to the single development machine and not easily accessed by other members of the integration team. A work-around solution to this problem was to use a PC database (FileMaker Pro) to store and manipulate CDMS requirements data. The expert system reads data exported from FileMaker and performs analyses and validity checks. We expect to transfer the data from FileMaker files to Oracle when the networked Oracle server hardware and software is installed.

In addition to CDMS, TBE has efforts underway to use expert Systems in several other areas of the payload integration process. One effort worth noting is the Functional Objectives Requirements Collection System (FORCS) which will be used to help PIs define functional objectives for their experiment. The current FO process suffers from many of the same problems as CDMS including the task of entering paper inputs from the PI, inconsistency in the way those requirements are stated, multiple (often inconsistent) copies of the data spread across disciplines, etc. The FO Expert System and the CDMS acquisition system are both parts of a larger effort to automate the acquisition of all experiment requirements and to store those requirements in a relational database for analysis and integration. The lessons learned in both these efforts will be applied to other

problems within the mission payload integration process.

REFERENCES

Clancey, William J.(1988). "The Knowledge Engineer as Student: Metacognitive Bases for Asking Good Questions" in Learning Issues for Intelligent Tutoring Systems, Springer Verlag

Craig. F.G., Cutts, D.E., Fennel, T.R. & Purves, B. (1990). "Graphical Explanation in an Expert System for Space Station Freedom Rack Integration", Fifth Conference on Artificial Intelligence for Space Applications, Huntsville AL, May

Doyle, John (1981). "A Truth Maintenance System" in Readings in Artificial Intelligence, pp 496-516 (Tioga Publ Co. 1981)

Fennel T.R. & Johannes, J. D. (1990). "An Architecture for Rule Based System Explanation" Fifth Conference on Artificial Intelligence for Space Applications, Huntsville AL, May 1990.

Gaines, B.R.(1989) "Integration issues in Knowledge Support Systems" in International Journal of Man-Machine Studies (1989) 31:5

Gruber, Thomas & Cohen, Paul (1987). "Principles of Design for Knowledge Acquisition" in Proceedings of 3rd IEEE Conference on AI Applications (1987)

Kitto, Catherine M. & Boose, John H. (1989). "Selecting Knowledge Acquisition Tools & Strategies based on Application Characteristics" in International Journal of Man-Machine Studies (1989) 31:8

MDC (1991). MDC G6854D Volume II, Appendix D, CDMS Forms and Instructions

Swigger, Kathleen M.(1989). "Managing Communication Knowledge" in Proceedings of the Second Intelligent Tutoring Systems Research Forum, San Antonio, TX (April 1989)

Wexelblat, Richard L. (1989). "On Interface Requirements for Expert Systems" in AI MAGAZINE 10:3 (Fall 89)